DEVELOPMENT OF OPTIMUM DESIGN METHODOLOGY FOR CONTINUOUS–TIME SIGMA-DELTA ANALOG-TO-DIGITAL CONVERTERS

An Undergraduate Research Scholars Thesis

by

YONG CHAN KIM

Submitted to Honors and Undergraduate Research
Texas A&M University
In partial fulfillment of the requirements for the designation as

UNDERGRADUATE RESEARH SCHOLAR

Approved by
Research Advisor: Dr. Aydin I. Karsilayan

May 2013

Major: Electrical Engineering

TABLE OF CONTENTS

Page

ABSTRACT

Development of Optimum Design Methodology for
Continuous-Time Sigma-Delta Analog-to-Digital Converters. (May 2013)

Yong Chan Kim
Department of Electrical and Computer Engineering
Texas A&M University

Research Advisor: Dr. Aydin I. Karsilayan
Department of Electrical and Computer Engineering

Today's wireless applications have demanded low-power, high-speed, and high-resolution analog-to-digital converters (ADCs) for the reductions in cost, component count, and product size. Continuous-time (CT) Sigma-Delta ADCs are well suited for satisfying the demands due to their better power efficiency, inherent anti-aliasing filtering, and high-speed properties. This thesis provides overviews of fundamentals with the focus on CT Sigma-Delta ADCs and investigates the loop filter design methodologies for CT low-pass sigma-delta (ADCs). The existing design methodologies for continuous-time low-pass sigma-delta analog-to-digital converters (ADCs) are presented. The drawbacks of the current methodology to obtain CT (s-domain) transfer function for a CT Sigma-Delta modulator are investigated. An optimized loop filter design methodology of CT low-pass Sigma-Delta ADCs using Matlab software is introduced in this work. A feed-forward topology is selected for the implementation of the CT Sigma-delta modulator. Two Matlab functions are presented. First function, named ctlooft, synthesizes a loop filter for a CT Sigma-Delta modulator, and the other, called ffcoeff, calculates the parameter coefficients for feed-forward implementation of CT modulator.

ACKOWLEDGMENTS

I sincerely appreciate my advisor, Dr. Aydin Karsilayan, for giving me an opportunity to conduct my undergraduate research under his guidance. I also thank him for providing me with detailed knowledge for the research. His electronic course greatly increased my interests in Analog and Mixed signals circuits. I sincerely appreciate his support of my academic endeavors.

I would like to take this opportunity to thank Dr. Alex Sprintson for maintaining Undergraduate Research Award scholarship. I also thank Dr. Scott Miller and Mr. Matthew Pariyothorn for supporting my undergraduate research by organizing and leading Research Opportunities for Engineering Scholars Program. Moreover, I appreciate Dr. Duncan Mackenzie and Mrs. Tammis Sherman for supporting and overseeing the Undergraduate Research Scholars Program.

NOMENCLATURE

AAF	Anti-Aliasing Filter
ADC	Analog-to-Digital Converter
BP	Band-Pass
CT	Continuous-Time
DAC	Digital-to-Analog Converter
DR	Dynamic Range
DT	Discrete-Time
ENOB	Effective Number of Bits
LP	Low-Pass
NTF	Noise Transfer Function
OBG	Out-of-Band Gain
OFDM	Orthogonal Frequency-Division Multiplexing
OSR	Oversampling Ratio
ΣΔ	Sigma-Delta
S/H	Sample and Hold
SNR	Signal-to-Noise Ratio
SQNR	Signal-to-Quantization Noise Ratio
STP	Signal Transfer Function
t	Time

CHAPTER I

INTRODUCTION

Motivation

As the “system on a chip”-style of design and manufacturing became prevalent in the late 1990s, electronic products in wireless applications can be made smaller and cheaper by increasing the integration level in all system functions onto a single substrate only with few external components. The integration task can be done more easily if signals are in digital form for on-chip process because digital signals are less vulnerable to corruption by circuit noise and process variations. Therefore, a shifting analog-to-digital converter (ADC) towards the antenna side of the receiver is important since it enables more digital integration of analog functions on a single digital chip. This trend made ADC to play an important role in the on-chip integration.

The technological advance in the wireless communication area has driven higher data rates become possible for broadband networks, which require high-performance circuit blocks. For example, a typical 54Mb/s transceiver transfers data using a channel bandwidth of 20MHz by applying orthogonal frequency-division multiplexing (OFDM) modulation scheme, so this type of broadband communication requires a wide bandwidth, high-speed, and high resolution analog-to-digital converter [1].

Conventional flash or pipeline ADCs operating at the Nyquist rate are not suitable for this case, since they do not provide the required speed at reasonable power dissipation. On the other hand, oversampling sigma-delta (ΣΔ) ADCs offer the most effective speed-accuracy tradeoff for wide signal bandwidths, making them suitable for wireless communications applications, especially for multi-standard implementations. Among the two types of ΣΔ ADCs, discrete-time (DT) ΣΔ converter is implemented using switched-capacitor techniques, and its bandwidth is generally limited to low frequencies as the settling time requirements for the charge transfer between the switched-capacitor integrators boost its power consumption [2-3]. The bandwidth of continuous-time (CT) ΣΔ ADC, on the contrary, is usually not limited by settling requirements; therefore, CT ΣΔ ADCs allow higher sampling frequencies. They also have the advantages of having no sample-and-hold in the front-end, and inherent anti-aliasing filter, as well as lower thermal noise generation by the filter circuits. As a result, these advantages lead to lower power consumption and smaller chip area. Therefore, CT ΣΔ ADCs are well suited for satisfying the demands of broadband communication systems due to their better power efficiency, inherent anti-aliasing filtering, high-speed properties, and CT ΣΔ ADCs are commonly used in modern communication systems as a part of wireless and wire-line receivers. However, currently available design methodology for CT ΣΔ ADCs does not take some practical limitations into consideration, so it results in ideal transfer functions that are not realizable. The method also requires further modifications of the transfer function for the practical implementation, and it causes degradation in SQNR. Therefore, the development of an optimum design methodology for CT ΣΔ ADCs is necessary because it will make the design process easier and produce more accurate design performance.

This paper provides overviews of fundamentals with the focus on CT ΣΔ ADCs and investigates the loop filter design methodologies for CT low-pass (LP) ΣΔ ADCs. Moreover, this thesis involves the development of optimum design methodologies for CT ΣΔ ADCs by determining the optimum loop-filter transfer function to obtain the maximum signal-to-quantization noise ratio (SQNR). Faster and more efficient CT ΣΔ ADC designs will be presented.

ΣΔ analog-to-digital converters

The history of ΣΔ modulator began in 1954 as Cutler filed a patent of a feedback system with a low-resolution quantizer in the forward path. The quantization error of a quantizer was fed back and subtracted from the input signal. This principle of improving the resolution of a quantizer by using a feedback and taking differences between the quantization error and the input signal became the fundamental concept of a sigma-delta converter [4]. In 1962, Inose et al. [5] presented the idea of adding a filter in the forward path of a delta modulator in front of the quantizer. This system was called a 'delta-sigma modulator or sigma-delta modulator', where 'delta' refers to the delta-modulator, and 'sigma' refers to summation by the integrator. The delta modulator is a feedback loop, containing an internal low-resolution ADC and DAC, as well as a loop filter. This structure can allow larger input signals, but the loop filter in the feedback path limit the achievable linearity and accuracy due to its nonidealities. A DAC and a demodulation filter are required in the demodulator. Since the filter has a high gain in the signal band, it not only amplifies the nonlinear distortion of the DAC but also picks up noise from the signal between the modulator and demodulator. A ΣΔ modulator shown in Fig. 1 avoids the shortcomings of the delta modulator. A ΣΔ modulator consists of a loop filter, a quantizer, and a DAC. The loop filter has a high gain in the signal band, so the in-band quantization noise is

decreased by the filter. The digital output contains a delayed analog input signal and differentiated version of the quantization error.

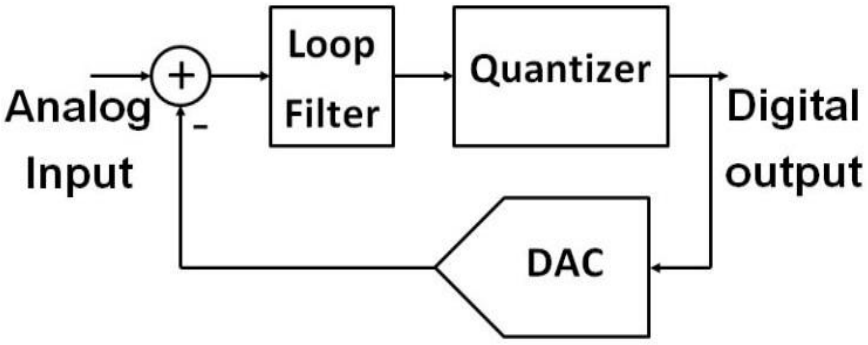

Figure 1. Sigma-Delta (ΣΔ) modulator used as an ADC

The ΣΔ ADCs are classified into two types based on the location of the sample-and-hold (S/H) block. If the sampling operation is performed outside a ΣΔ loop, its loop filter is a z-domain transfer function, and the ΣΔ ADC is called a DT ΣΔ ADC. If the sampling is occurred inside the loop, it has a s-domain loop filter, and the ΣΔ ADC is referred as CT ΣΔ ADC. The block diagrams of DT and CT ΣΔ ADCs are shown in Fig. 2(a) and Fig. 2(b) respectively. As shown in Fig. 2(a) DT ΣΔ ADC requires an additional anti-aliasing filter to avoid signals separated by a multiple of the sampling frequency. On the other hand, CT ΣΔ ADC has an inherent property of antialiasing so that it does not employ a separate anti-aliasing filter.

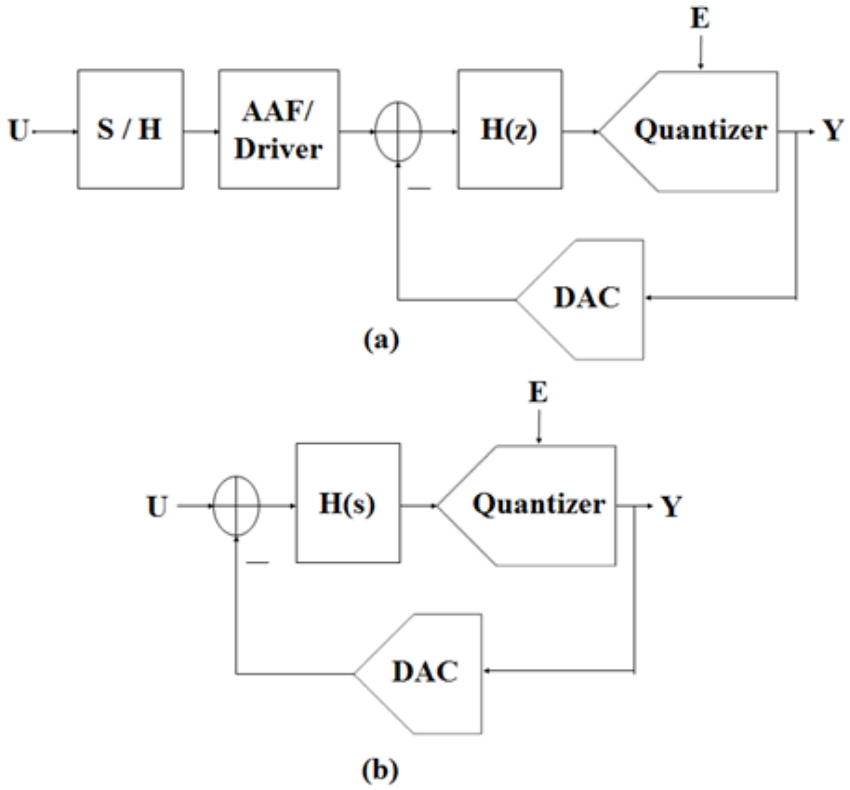

Figure 2. Two types of ΣΔ ADC (a) DT ΣΔ ADC and (b) CT ΣΔ ADC

The majority of ΣΔ ADCs in the literature are implemented as DT circuits such as switched-capacitor (SC) circuits in last decades due to mature design methodologies and robustness. In recent years, CT ΣΔ ADCs became to receive more attentions from researchers in wireless applications since CT ΣΔ ADCs have the advantages comparing to DT ΣΔ ADCs such as low power consumption, high sampling speed and inherent anti-aliasing filter; therefore, they can be used to extend battery life and to simplify system complexity. DT ΣΔ ADCs can be easily implemented in MATLAB software by using the Delta Sigma toolbox developed by Dr. Richard Schreier. There are, on the other hand, no matured design methodologies for designing CT ΣΔ ADCs yet, so the development of an optimum design methodology for CT ΣΔ ADCs is necessary. Existing design methods for CT circuits are not sufficient to provide the most optimum solution.

With this research, faster and more efficient ADC designs will be feasible. The goal is to develop the optimum design methodology resulting in an improved toolbox for the design of CT ΣΔ ADCs. The toolbox developed at the end of this research will be useful for numerous circuit designers working in IC design area.

Organization

This thesis consists of total four chapters. First chapter discusses motivation of the thesis and provides a brief overview of ΣΔ ADCs. Several reasons why CT ΣΔ ADCs are chosen for this thesis are first described in Chapter I. A history of ΣΔ ADCs and the differences between CT and DT ΣΔ ADCs are provided.

Chapter II introduces fundamental knowledge of ΣΔ ADCs and the importance of noise transfer function.

Chapter III summarizes the existing loop filter design methodologies for CT ΣΔ ADCs with providing the shortcomings of the design methods in a sample design with typical specifications. The optimum loop filter design methodology is explained. This chapter also presents implementation of the developed loop filter design methodology for CT ΣΔ ADCs at the system levels.

Chapter IV discusses overall summaries, conclusions, and future work.

CHAPTER II

OVERVIEW OF ΣΔ ANALOG-TO-DIGITAL CONVERTERS

ΣΔ analog-to-digital conversion

An analog-to-digital converter samples the continuous input signal at discrete time and quantizes the amplitude of the input in discrete amplitude levels. Since the number of quantization levels is finite, a quantization error always exists; and this limits the achievable resolution. A ΣΔ modulator can improve the resolution of the ADC due to its oversampling and noise-shaping properties. Oversampling is the sampling rate higher than Nyquist frequency which is twice the signal bandwidth. The oversampling ratio (OSR) is defined as

$$\text{OSR} = \frac{f_s}{2f_b} \tag{1}$$

where f_s is the sampling rate and f_b is the signal bandwidth. Noise-shaping is the process of filtering of the quantization errors and shaping their frequency response. This process reduces the quantization error power in the frequency band of interest and increases the quantization error power outside that band, so higher accuracy can be achieved in a small bandwidth. As discussed in chapter I, a single loop ΣΔ ADC consists of a loop filter, a low resolution quantizer, and a feedback digital-to-analog converter (DAC). The quantizer modulates the continuous input signal into a discrete signal by approximating the analog input well in a desired frequency range. The quantization of the continuous signal produces the noise, and the loop filter shapes the quantization noise.

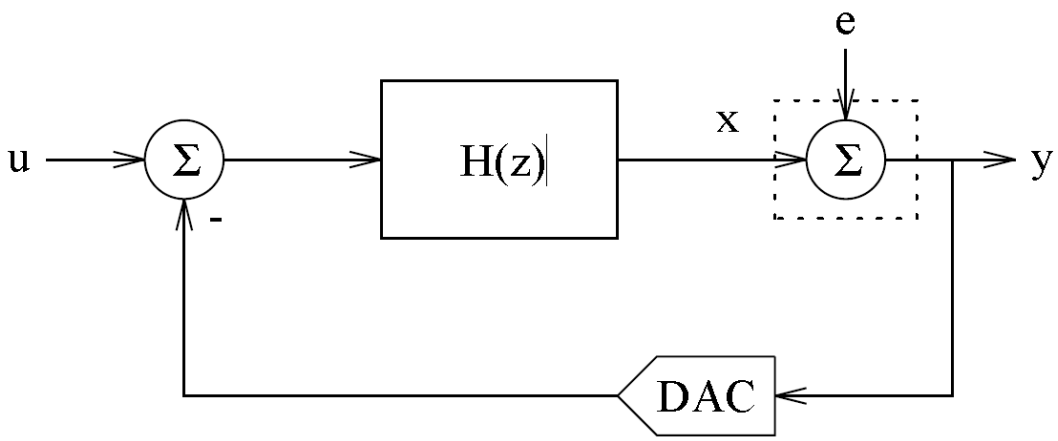

Figure 3. Linearizing the quantizer in a ΣΔ modulator [6].

As shown in Fig. 3, the quantizer can be modeled as an adder for the loop filter output signal and the quantization error, an input e. The output y can be expressed in terms of u and e as

$$\mathrm{Y}(z) = \frac{H(z)}{1 + H(z)} U(z) + \frac{1}{1 + H(z)} E(z) \tag{2}$$

Where $U(z)$ is the input signal and $E(z)$ is the quantization noise. The signal transfer function (STF) and the noise transfer function (NTF) can be written as

$$\mathrm{STF} = \frac{H(z)}{1 + H(z)} \text{ and } \mathrm{NTF} = \frac{1}{1 + H(z)} \tag{3}$$

The zeros of NTF become the poles of H(z). If H(z) >> 1,

$$\mathrm{Y}(z) \approx \mathrm{U}(z). \tag{4}$$

Therefore, as long as the loop filter is great, the output spectrum is almost the same as the input signal. The output Y of Fig.3 is also equal to

$$\mathrm{y}(n) = \mathrm{u}(n-1) + \mathrm{E}(n) - \mathrm{E}(n-1) \tag{5}$$

The modulator passes the unchanged input signal with one sampling period delay and differentiates the quantization error. The average of the quantization errors decreases as the oversampling ratio increases. This statement will be further explained in the next chapter.

The importance role of noise transfer function (NTF)

A ΣΔ modulator generally employs a NTF in a high pass form, and it plays a very important role in any sigma-delta modulator since it is related to the main component of the ΣΔ modulator, the loop filter, as

$$\mathrm{H(z)} = \frac{1}{NTF} - 1 \qquad (6)$$

There are two different ways to design NTF according to the location of the zeros and poles of the filter. First type of a NTF is to locate coincident zeros in the middle of the signal band in order to attenuate the in-band noise. The other type is a NTF with optimized zeros. Its zeros are spread across the band of interest to maximize the in-band attenuation. Since the noise shaping of the ΣΔ modulator mainly depends on how the NTF is designed, designing a NTF is a very crucial step in an optimum design of ΣΔ ADC. In the next chapter, the optimization of a NTF will be further examined.

CHAPTER III

METHODS

The hand calculation method for designing a CT ΣΔ loop filter

Designing the NTF is the first and most important stage of the loop filter design for a CT ΣΔ ADC. There are several loop filter design methodologies for CT ΣΔ ADCs. The most common method to design a CT ΣΔ loop filter is to first find the equivalent DT ΣΔ loop filter and then transform it to CT using impulse invariant transformation [4]. Another method is to design a CT loop filter as a starting point instead of matching a CT loop filter to a DT loop filter. The CT loop filter is further optimized for its robustness, and then it is simulated in DT to assess the final performance in terms of stability and SNR. In this work, the first methodology, by using the impulse-invariant transformation to convert DT ΣΔ loop filter to CT ΣΔ loop filter, is investigated and optimized. The hand calculation method is to implement the following algorithm [4], [7], and [8]:

1) To meet the resolution specification, choose the variables for system-level design such as oversampling ratio (OSR), number of levels in the internal quantizer (N), order of the loop filter (L) based on the following equations [8] :

$$\text{Effective Number of Bits (ENOB)} = \frac{SQNR - 1.76}{6.02} \tag{7}$$

$$\text{SQNR}_{max}\ (\text{in dB}) = (20\text{L} + 10)\log_{10} OSR - 10\log_{10}\left(\frac{\pi^{2L}}{2L+1}\right) + 6.02N + 1.76 \tag{8}$$

$$\text{Dynamic Range (DR)} = \frac{3}{2}\left(\frac{2L+1}{\pi^{2L}}\right)(2^N - 1)^2 OSR^{2L+1} \tag{9}$$

The equation (8) and (9) are valid only if the noise transfer function is in the form of $(1-z^{-1})^L$, which is not true for most wideband and high resolution CT ΣΔ ADCs. It does provide some guidance on selecting OSR, N, and L.

2) Choose a high-pass filter transfer function for the NTF among Butterworth, Chebyshev, and Inverse-Chebyshev. The resulting transfer function should be in the form as the following:

$$\mathrm{H(z)} = \frac{b_1 + b_2 z^{-1} + \cdots + b_{n+1} z^{-n}}{1 + a z^{-1} + \cdots + a_{n+1} z^{-n}} \tag{10}$$

The (10) need to satisfy the condition, $\mathrm{H}(\infty) = 1$. The condition can be achieved by dividing the (10) by b_1. The desired signal bandwidth can be a starting point for the stop-band of e NTF. To obtain optimum SQNR, the zeros of the transfer function can be spread across the signal bandwidth.

3) Predict the stability of the ADC by simulating it. Determine its maximum stable input and peak SNR.
4) If the ADC is unstable, reduce the out-of-band gain (OBG) of the NTF by lowering the cutoff frequency.
5) If the ADC is stable but the SQNR is not adequate, increase the OBG of the NTF.
6) Use equation (6) to obtain the DT loop transfer function from the NTF.
7) Modify the loop transfer function to account for excess loop delay by factoring out z^{-L} term, where L is the amount of delay in the loop.
8) Find the equivalent CT ΣΔ loop filter based on the DT ΣΔ loop filter by means of the impulse-invariance transform taking into account the shape of the DAC feedback pulse shown in the following equation:

$$Z^{-1}[H(z)] = L^{-1}[H_{DAC}(S) * H(s)] \tag{11}$$

where Z^{-1} and L^{-1} stand for inverse Z and inverse Laplace transforms respectively and $H_{DAC}(S)$ is the frequency response of the DAC in the feedback loop. In the time domain, (11) becomes as $\mathrm{h(n)} = \{h_{DAC}(t) * h(t)\}|_{t=nT_s}$, where h(n), $h_{DAC}(t)$, and $h(t)$ represent the impulse responses of the DT loop filter, the DAC, and the CT loop filter respectively. Hand calculation method is vulnerable to errors, and it is a time consuming as steps 2-5 are repeated.

The method using Computer-Aided-Design (CAD) tools

The Delta-Sigma Toolbox developed by Dr. Richard Schreier is a useful free MATLAB toolbox for designing ΣΔ modulators [9]. One of the most popular functions of the tool box is synthesizeNTF. The function takes input arguments such as order, OSR, number of quantization levels, optimization flag, and center frequency of the target system, and it generates a DT NTF as an output. First five steps of the algorithm used in the hand calculation method can be easily done by using the function. Once the NTF is obtained, the rest steps of the designing process are the same as the hand calculation since there is no MAtlab function that produces CT ΣΔ loop filter. The rest steps can also be done in Matlab. If the order of the CT modulator is a first order, this method works fine. However, 2nd order- or higher order-CT ΣΔ loop filter obtained by using this method is often not realizable. For example, if a second-order, an OBG of 3, 3bit (with 7 quantization output levels) topology is chosen with an OSR of 16 for input arguments for the synthesizeNTF function, the NTF is generated as the following:

$$\mathrm{NTF(z)} = \frac{z^2 - 1.987z + 1}{z^2 - 0.2596z + 0.06942} \tag{12}$$

The DT loop transfer function of the NTF is

$$\mathrm{H(z)} = \frac{1.7275z(z - 0.5387)}{(z^2 - 1.987z + 1)} \tag{13}$$

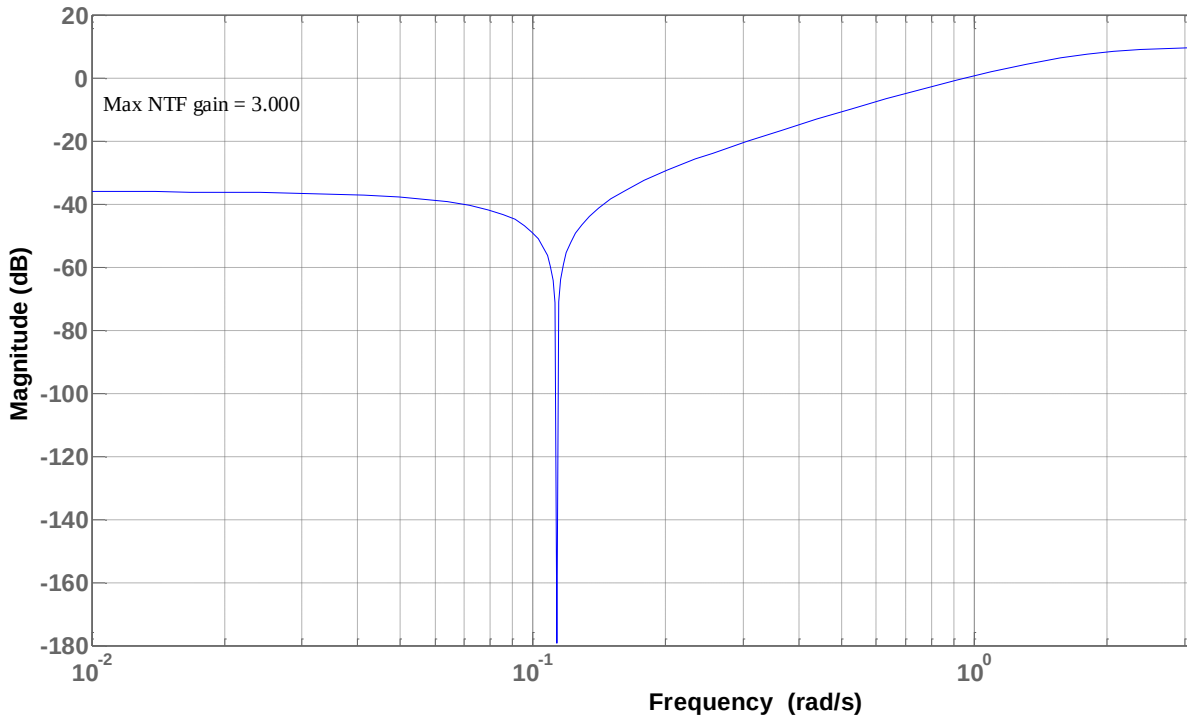

Figure 4. Bode plot of the NTF using synthesizeNTF function.

The bode plot of the NTF using synthesizeNTF fuction is shown in Fig.4. The Q factor value of the NTF looks very high as it has a very narrow and sharp window. If the CT ΣΔ ADC has a bandwidth of 20MHz, the equivalent CT loop filter is obtained as the following:

$$\mathrm{H(s)} = \frac{1.9672(s^2 + 1.472 \times 10^8 s + 6.65 \times 10^{15})}{(s^2 - 4.424 \times 10^{-9} s + 1.851 \times 10^{14})} \tag{14}$$

However, the equation (14) is not realizable if this loop function is implemented by using a biquad filter employing feed-forward architecture. The coefficient, -4.424×10^{-9}, from the denominator of (14) is not possible value since the value determined by $\frac{\omega_o}{Q}$, where ω_o is the center angular frequency of input signal and Q is the quality factor of the biquad filter, cannot be a negative value. Since the denominator of (14) is obtained from the numerator of (13) using the equation (6), the numerator of (13) which are zeros of NTF requires a further modification.

In order to obtain a realizable CT ΣΔ loop filter, the zeros of the ideal NTF synthesized by using the Delta-Sigma toolbox need to be optimized, but the poles of the ideal NTF do not need any modification and can be used for the optimized NTF. The optimization method is explained in the following example. A 2nd order- CT ΣΔ loop filter has the transfer function in the form as [10]:

$$\mathrm{H(s)} = \frac{K(c_1 \frac{\omega_{01}}{Q} s + c_2 {\omega_{01}}^2)}{s^2 + \frac{\omega_{01}}{Q} s + {\omega_{01}}^2}$$

If the loop filter is implanted in a feed-forward topology by using a biquad filter, k is the low frequency gain. The coefficient c_1 and c_2 are the feed-forward coefficients realized as ratio of resistors. ω_o and Q are cut-off frequency and the quality factor of the biquad filter. The equivalent DT NTF of the CT ADC shown in Fig. 2 (b) is the following:

$$\mathrm{NTF(z)} = \frac{1}{1 + Z[H_{DAC}(S) * H(s)]} = \frac{1}{1 + H(z)}$$

$H(z)$ is the open-loop transfer function of the corresponding DT ADC. To map from the NTF of the CT ADC to the DT ADC, $Z[H_{DAC}(S) * H(s)]$, the z-transform of $H_{DAC}(S) * H(s)$, should be equal to $H(z)$. The equivalent DT NTF of the 2nd order- CT ΣΔ ADC can be derived as the following [10]:

$$\mathrm{NTF(z)} = \frac{z^2 - 2\cos(\beta\omega_0 T_s)e^{\alpha\omega_0 T_s} z + e^{2\alpha\omega_0 T_s}}{d_1 z^2 + d_2 z + d_3} \quad (15)$$

where $\alpha = -\frac{1}{2Q}, \beta = \sqrt{1 - \frac{1}{4Q^2}}, \omega_o = 2\pi \cdot f_{in}, and\ T_s = \frac{1}{f_s}$.

Since only zeroes of the NTF by generated by the Delta-Sigma function are needed to be optimized, the poles of NTF can be used for designing the equivalent CT ΣΔ loop filter. If a 2nd

order- DT ADC is chosen for the DT NTF, the synthesizeNTF function produces its output as the following [11]:

$$\mathrm{NTF(z)} = \frac{z^2 + cz + 1}{z^2 + d_2 z + d_3} \tag{16}$$

Comparing (15) to (12) and (16), the values of α and β from (15) are equal to zero and one respectively, so Q value used in the Delta-Sigma synthesizeNTF function is infinite for the DT modulator. However, the actual Q value is finite, so the numerator of the NTF needs a modification. As shown in Fig. 4, the location of zeros is the same as where the minimum magnitude of the NTF is. Therefore, if the frequency ($w_{min} = \beta\omega_0 T_s$) of the conjugate zeros is known, the numerator of (15) can be written as the following:

$$z^2 - 2\cos(\omega_{min}) e^{-\frac{\omega_{min}}{\sqrt{4Q^2-1}}} z + e^{-\frac{2\omega_{min}}{\sqrt{4Q^2-1}}}$$

The frequency that gives the minimum magnitude can be easily obtained by using two Matlab functions such as bode and sort. Bode function provides the response magnitudes, phases, and the frequency vectors. Sort function is used to sort the response magnitudes in ascending order. Then, the first element of the sort output holds the minimum magnitude. The frequency vector pointing the minimum magnitude is ω_{min}.

In order to obtain the optimized DT NTF, the value of Q factor needs to be determined at the beginning of the NTF design. This will ensure the DT NTF to be realizable. Once the realizable DT NTF is obtained, the CT ΣΔ loop filter can be obtained by following the rest steps of the aforementioned algorithm. The optimized DT NTF using the same specifications used earlier with choosing Q factor of 3 is the following:

$$\mathrm{NTF(z)} = \frac{z^2 - 1.949z + 0.9624}{z^2 - 0.2596z + 0.06942}$$

The bode plot of the modified NTF is shown in Fig. 5. The zero location of the modified NTF is the same as the location of the original NTF. The resulting CT ΣΔ loop filter is realizable as follows:

$$\mathrm{H(s)} = \frac{1.6898(s^2 + 8.027 \times 10^8 s + 1.971 \times 10^{17})}{(s^2 + 2.453 \times 10^7 s + 5.414 \times 10^{15})}$$

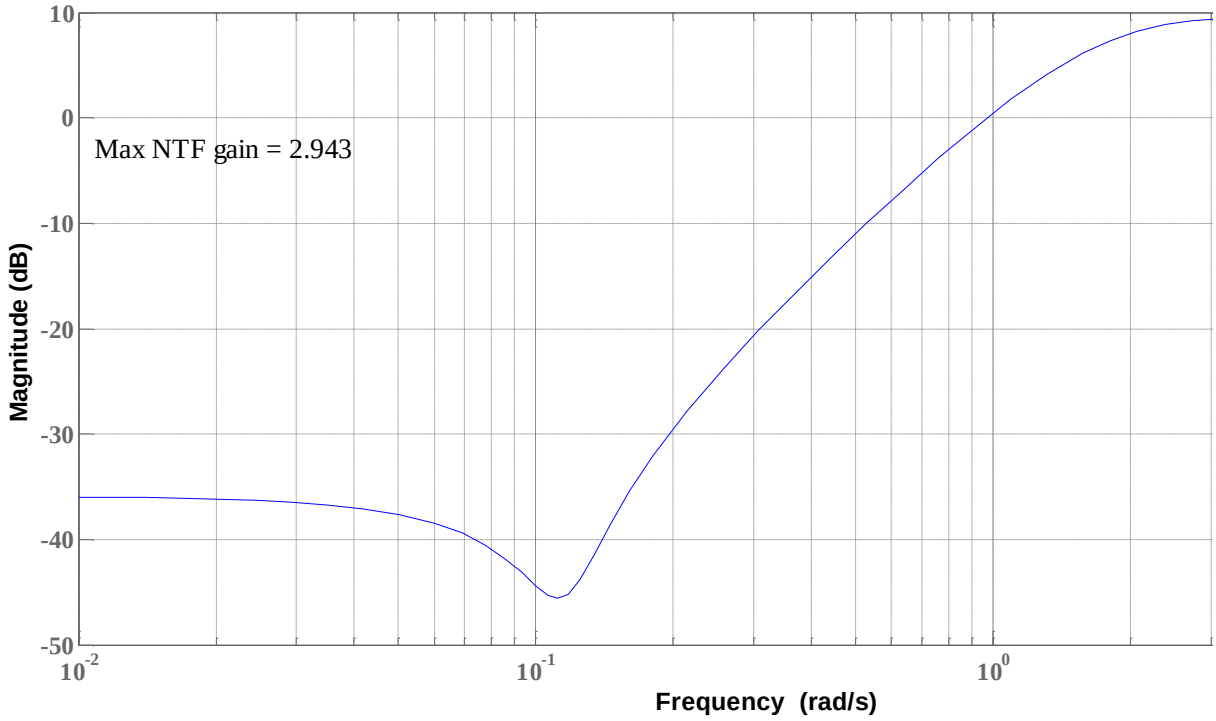

Figure 5. Bode plot of the NTF with Q value modified to 3.

Loop filter implementation

CT ΣΔ ADCs have two popular architectures: feedback and feed-forward architectures. Comparing to feed-forward architecture, feedback architectures does not require a large adder before the quantizer. However, the feed-forward topology is preferred in this work due to its power efficient design. The input swing at the internal nodes of the loop is relaxed in case of feed-forward architecture so that this significantly reduces the power consumed by the non-

critical blocks and the power requirement of the first stage of the loop. In this subsection, the feed-forward implementations of 3rd order- and 5th order-CT low-pass ΣΔ ADCs are presented.

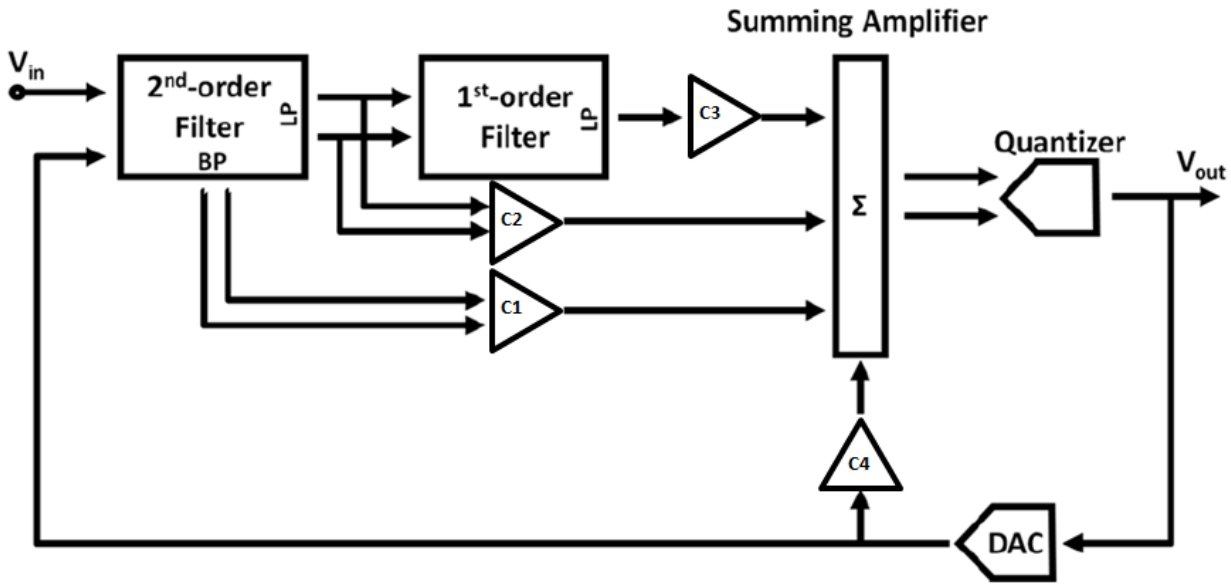

Figure 6. Feed-forward architecture of the 5th order- **ΣΔ** ADC [12]

In particular, the feed-forward implementation of 3rd order- CT low-pass $\Sigma\Delta$ ADCs is investigated in the following paragraph. The system architecture of the proposed feed-forward work is shown in Fig. 6 [12]. Its loop filter consists of a biquad filter and one lossy-integrator. As a feed-forward topology, the summing amplifier is employed before a 3-bit quantizer which provides 7 quantization levels from the analog input. A DAC produces two different feedback paths, one (c_4) for stability and another for main feedback. The general form of 3rd order- CT $\Sigma\Delta$ loop filter is the following:

$$\mathrm{H(s)} = \frac{a_1 s^3 + a_2 s^2 + a_3 s + a_4}{s^3 + b_1 s^2 + b_2 s + b_3} \tag{17}$$

C_4, which is the coefficient for the fast feedback path, is equal to a_1 of the numerator of (17). After subtracting C_4 from (17), the transfer function can be expressed as:

$$H'(s) = \frac{n_1 s^2 + n_2 s + n_3}{(s^2 + d_1 s + d_2)(s + d_3)} \tag{18}$$

The transfer function for the proposed feed-forward work is the following:

$$H'(s) = \frac{K_1\left(C_1 \frac{\omega_{o1}}{Q} + C_2 {\omega_{o1}}^2\right)}{s^2 + \frac{\omega_{01}}{Q} s + {\omega_{01}}^2} + \frac{K_1 {\omega_{o1}}^2}{s^2 + \frac{\omega_{01}}{Q} s + {\omega_{01}}^2} \times \frac{K_1 C_3 \omega_{o2}}{s + \omega_{o2}} \tag{19}$$

By comparing the denominator of (19) to the denominator of (18), the values of ω_{01}, ω_{02}, and Q are easily calculated, but K_1, C_1, C_2, and C_3 cannot be recognized separately, only the products of K_1C_1, K_1C_2, and K_1C_3 are known. Therefore, K_1 needs to be chosen to reasonable value, and then C_1, C_2, C_3, and C_4 can be calculated.

The feed-forward implementation of 5th order- CT low-pass ΣΔ ADCs is completed in a similar way as in the 3rd order case. The fifth order loop filter transfer function can be realized by using the feed-forward architecture shown in Fig. 9.

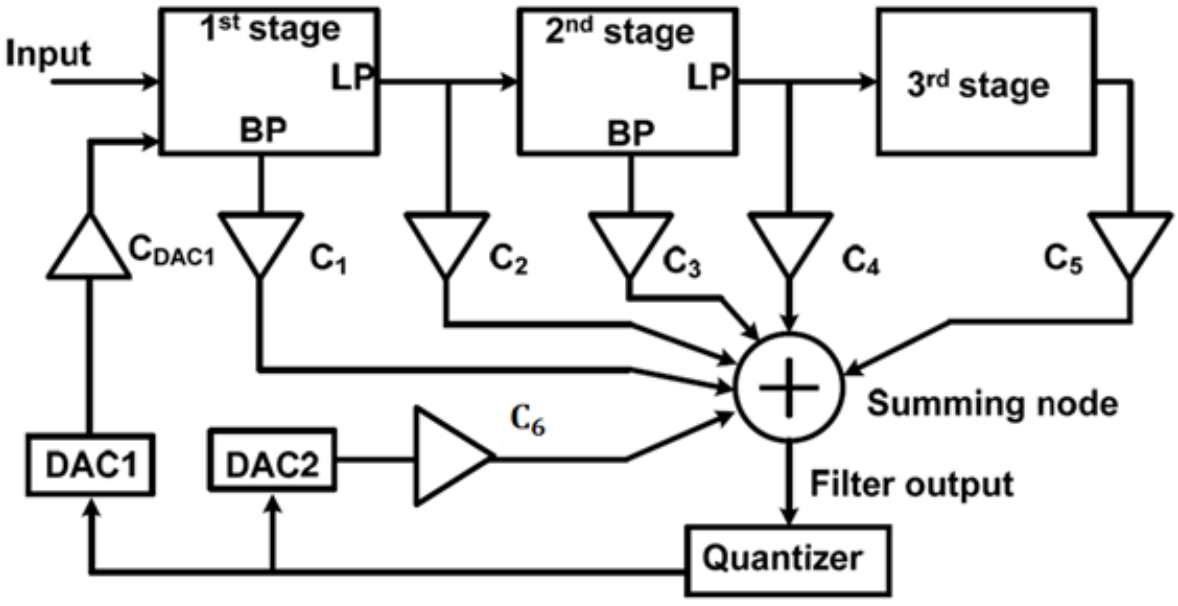

Figure 7. Feed-forward architecture of the 5th order- **ΣΔ** ADC [13]

This topology employs two biquad filters and one integrator to realize the loop transfer function. The summing node is also required to sum up the outputs of the each stage. The two DAC are used in the feedback path. The general form of 5th order- CT ΣΔ loop filter is the following:

$$\mathrm{H(s)} = \frac{a_1 s^5 + a_2 s^4 + a_3 s^3 + a_4 s^2 + a_5 s + a_6}{s^5 + b_1 s^4 + b_2 s^3 + b_3 s^2 + b_4 s + b_5} \qquad (20)$$

As the fast feedback coefficient was obtained in the 3rd order case, the value of a_1 is equal to C_6. Subtracting C_6 from (20), the transfer function becomes as:

$$\mathrm{H'(s)} = \frac{n_1 s^4 + n_2 s^3 + n_3 s^2 + n_4 s + n_5}{(s^2 + d_1 s + d_2)(s^2 + d_3 s + d_4)(s + d_5)} \qquad (21)$$

The transfer function of the loop filter can be realized by the following:

$$\mathrm{H'(s)} = \frac{K_1\left(C_1 \frac{\omega_{o1}}{Q_1} + C_2 {\omega_{o1}}^2\right)}{s^2 + \frac{\omega_{01}}{Q_1} s + {\omega_{01}}^2} + \left(\frac{K_1 {\omega_{o1}}^2}{s^2 + \frac{\omega_{01}}{Q_1} s + {\omega_{01}}^2}\right)\left(\frac{K_2\left(C_3 \frac{\omega_{o2}}{Q_2} + C_4 {\omega_{o2}}^2\right)}{s^2 + \frac{\omega_{02}}{Q_2} s + {\omega_{02}}^2}\right)$$

$$+ \left(\frac{K_1 {\omega_{o1}}^2}{s^2 + \frac{\omega_{01}}{Q_1} s + {\omega_{01}}^2}\right)\left(\frac{K_2 {\omega_{o2}}^2}{s^2 + \frac{\omega_{02}}{Q_2} s + {\omega_{02}}^2}\right)\left(\frac{K_3 C_5 \omega_{o3}}{s + \omega_{o3}}\right) \qquad (22)$$

The parameter of (22) is calculated based on the (21) in a same manner completed earlier.

CHAPTER IV

RESULTS AND CONCLUSIONS

A Matlab function, ctloopft, is developed. It takes input parameters such as the order of the loop filter, the oversampling ratio, the number of quantization levels, the maximum out-of-band gain, and the finite Q factor. The function provides the CT ΣΔ loop filter and the optimized NTF as a result. Its function manual is described in Appendix. Another Matlab function, ffcoeff, is also generated. This function takes the 3^{rd} or 5^{th} CT loop transfer function as an input and determines the coefficients for the modulator with feed-forward topology. Its function manual is also explained in Appendix.

In this work, an optimized loop filter design methodology for CT ΣΔ ADCs is proposed. The proposed method uses Matlab software to optimize the current design methodology and reduces the quite amount of time for designing a realizable CT loop transfer function. As a result of this work, two Matlab functions were developed. The ctloopft function synthesizes a CT loop filter, and the ffcoeff function determines the coefficients for implementing the CT modulator with a feed-forward architecture. Since the actual implementation of CT ΣΔ ADC at the system or transistor level has not completed, behavior simulation of a CT low-pass ΣΔ ADC using the proposed methodology in this work could lead to take more practical limitations into consideration as further modification of the transfer function may be required. Therefore, it will be a good starting point for future work.

REFERENCES

[1] M. Engels, *Wireless OFDM Systems*. Norwell, MA: Kluwer, 2002.

[2] I. Fujimori *et al.*, "A 90-dB SNR 2.5-MHz output-rate ADC using cascaded multi-bit delta-sigma modulation at 8_oversampling ratio," *IEEE. J. Solid-State Circuits*, vol. 35, pp. 1820–1828, Dec. 2000

[3] Y. Geerts, M. S. J. Steyaert, and W. Sansen, "A high-performance multi-bit delta sigma CMOS ADC," *IEEE J. Solid-State Circuits*, vol.35, pp. 1829–1840, Dec. 2000.

[4] Norsworthy, S.R., R. Schreier, and G.C. Temes, Delta-sigma data converters – Theory, design, and simulation, IEEE Press, New York, 1997.

[5] H. Inose, Y. Yasuda, and J. Murakami, "A telemetering system by code modulation-Δ-Σ modulation," IRE Trans. Space Electron. Telemetry, vol. SET-8, pp. 204-209, Sept. 1962

[6] J. A. Cherry and W. M. Snelgrove, "Continuous-Time Delta-Sigma High-Speed A/D Conversion," Kluwer Academic Publishers, Norwell, 2000.

[7] R. Schreier and G. Temes, Understanding Delta-Sigma Data Converters. New York: Wiley-IEEE Press, 2004.

[8] X. Chen, "A wideband low-power continuous-time delta-sigma modulator for next generation wireless applications," Ph.D. Thesis, Oregon State Univ., Corvallis, Oregon, Mar. 2007.

[9] R. Schreier, Delta-Sigma Toolbox. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=19& objectType=file.

[10] H. Lin, A. Motozawa, P. L. Re, K. Iizuka, H. Kobayashi, and H. San, "Study of Q Factor and Loop Delay Effects of Continuous-Time ΔΣ AD Modulator," Hunan: ASIC, 2009.

[11] F. M. Gardner, "A transformation for digital simulation of analog filters," *IEEE Trans. Commun.*, pp. 676–680, Jul. 1986.

[12] K. M. Kim, "A 3rd Order Continuous-Time Low-Pass Sigma-Delta (ΣΔ) Analog-to-Digital Converter for Wideband Applications," An Honors Fellows Thesis, Dept. Elect. Comput. Eng., Texas A&M Univ., College Station, Texas, 2011.

[13] J. Silva-Martinez, Fundamentals of Sigma-Delta Modulators Part 1.2. [Online]. Available: http://ame.tamu.edu/~jsilva/610/Lecture%20notes/Sigma-Delta-1.2.pdf.

APPENDIX A

MATLAB CODE

Function code: ctloopft

```
function [Hs ntf] = ctloopft(order,OSR,Nlev,Hinf,bw,qf_biquad1,qf_biquad2)
parameters = {'order' 'OSR' 'Nlev' 'Hinf' 'bw' 'qf_biquad1' 'qf_biquad2'};
defaults = {5 16 7 1.5 20e6 1 1};
opt=1;
f0=0;
form='CIFF';% CIFF=Cascade-of-integrators, feed-forward form.
Fs=OSR*2*bw;
Ts=1/Fs;
fprintf(1,'\t\t\t%dth-Order Lowpass\n\n',order);
%%%%% To obtain noise transfer function(NTF) generated by %%
%%using the Sigma-Delta Toolbox by Richard Schreier%%
if order==1
ntf1 = synthesizeNTF(order,OSR,1,Hinf);% synthesize a NTF for a SD modulator
[b1,a1] = tfdata(ntf1,'v');
end
if order==2
ntf1 = synthesizeNTF(order,OSR,1,Hinf);% synthesize a NTF for a SD modulator
% Get and modify the zeros of ntf
[ntf_mag ntf_phase ntf_w] = bode(ntf1);
```

```
[ntf_zero_mag ntf_zero_ix] = sort(ntf_mag);
ntf_minzero_freq = ntf_w(ntf_zero_ix(1));
%ntf_zero_ix(1) stores the minimum value of mag
%ntf_minzero_freq stores the location (frequency) of the zero
beta1 = ntf_minzero_freq; % frequency of the conjucate zero
[b1,a1] = tfdata(ntf1,'v');
alpha1 = -beta1/sqrt(4*qf_biquad1^2-1);
r1 = exp(alpha1);
b1 = [1 -2*r1*cos(beta1) r1^2];
end
if order==3
ntf1 = synthesizeNTF(order,OSR,1,Hinf);% synthesize a NTF for a SD modulator
% Get and modify the zeros of ntf
[ntf_mag ntf_phase ntf_w] = bode(ntf1);
[ntf_zero_mag ntf_zero_ix] = sort(ntf_mag);
ntf_minzero_freq = ntf_w(ntf_zero_ix(1));
%ntf_zero_ix(1) stores the minimum value of mag
%ntf_minzero_freq stores the location (frequency) of the zero
beta1 = ntf_minzero_freq; % frequency of the conjucate zero
[b1,a1] = tfdata(ntf1,'v');
alpha1 = -beta1/sqrt(4*qf_biquad1^2-1);
r1 = exp(alpha1);
b1 = conv([1 -1],[1 -2*r1*cos(beta1) r1^2]);
```

```
end
if order==4
ntf1=synthesizeNTF(order,OSR,opt,Hinf,f0); % synthesize a NTF for a SD modulator
% Get the zero
[ntf_mag ntf_phase ntf_w] = bode(ntf1);
[ntf_zero_mag ntf_zero_ix] = sort(ntf_mag);
ntf_minzero_freq = ntf_w(ntf_zero_ix(1));
ntf_minzero_freq2 = ntf_w(ntf_zero_ix(2));
%ntf_zero_ix(1) stores the minimum value of mag
%ntf_zero_ix(2) stores the second smallest value of mag
%ntf_minzero_freq stores the location (frequency) of the first zero
%ntf_minzero_freq2 stores the location (frequency) of the second zero
beta1 = ntf_minzero_freq; % frequency of conjucate zeros
beta2 = ntf_minzero_freq2;
[b1,a1] = tfdata(ntf1,'v');
alpha1 = -beta1/sqrt(4*qf_biquad1^2-1);
alpha2 = -beta2/sqrt(4*qf_biquad2^2-1);
r1 = exp(alpha1);
r2 = exp(alpha2);
b1 = conv([1 -2*r1*cos(beta1) r1^2],[1 -2*r2*cos(beta2) r2^2]);
end
if order==5
ntf1=synthesizeNTF(order,OSR,opt,Hinf,f0); % synthesize a NTF for a SD modulator
```

```
% Get the zero
[ntf_mag ntf_phase ntf_w] = bode(ntf1);
[ntf_zero_mag ntf_zero_ix] = sort(ntf_mag);
ntf_minzero_freq = ntf_w(ntf_zero_ix(1));
ntf_minzero_freq2 = ntf_w(ntf_zero_ix(2));
%ntf_zero_ix(1) stores the minimum value of mag
%ntf_zero_ix(2) stores the second smallest value of mag
%ntf_minzero_freq stores the location (frequency) of the first zero
%ntf_minzero_freq2 stores the location (frequency) of the second zero
beta1 = ntf_minzero_freq; % frequency of conjucate zeros
beta2 = ntf_minzero_freq2;
[b1,a1] = tfdata(ntf1,'v'); %numerator data -> b1
%denomenator data -> a1
alpha1 = -beta1/sqrt(4*qf_biquad1^2-1);
alpha2 = -beta2/sqrt(4*qf_biquad2^2-1);
r1 = exp(alpha1);
r2 = exp(alpha2);
b1 = conv([1 -0.95],conv([1 -2*r1*cos(beta1) r1^2],[1 -2*r2*cos(beta2) r2^2]));
end


ntf1 = filt(b1,a1,1); %digital filter H(z^-1)=b1(z^-1)./(a1(z^-1)
%with sampling time 1
```

```
ntf1 = zpk(ntf1);
[z,p,k]=zpkdata(ntf1,'v');
ntf=zpk(z,p,k,1);
% Plot NTF
ntf_mag = bode(ntf1,pi);
figure
bodemag(ntf1);
grid on;
s = sprintf('Max NTF gain = %4.3f \n',ntf_mag);
text(0.015,16,s);


%%%%% To obtain SNR values according to amplitudes %%%%%
N = 8192; % Number of points in the FFT
fB = ceil(N/(2*OSR)); % Signal bandwidth
f = floor(2/3*fB); % Input tone
amp1 = [-90:5:-15 -12 -10:2:-2 -1 0];
npoints = length(amp1);
snr1 = zeros(1,npoints);
maxsnr = snr1(1);
inp_maxsnr = amp1(1);
stability = 1;
maxstabin = 0;
for i=1:npoints
```

```
ampl = 10^(amp1(i)/20); % ampl = 10^(-3.5/20);
u = ampl*(Nlev-1)*sin(2*pi*f/N*[0:N-1]);
v = simulateDSM(u,ntf1,Nlev);
spec = fft(v.*ds_hann(N))/(N/4);
snr1(i) = calculateSNR(spec(1:fB),f);
if isinf(snr1(i))
maxsnr = 0;
snr1(i) = 0;
inp_maxsnr = amp1(i);
maxstabin = 0;
else
if snr1(i) > maxsnr
maxsnr = snr1(i);
inp_maxsnr = amp1(i);
end
end
if (i>3)
if (snr1(i)<(snr1(i-1)))&&(snr1(i-1)<(snr1(i-2)))&&(stability==1)&&(i>12)
stability = 0;
maxstabin = amp1(i-2);
end
end
end
```

```
figure
plot(amp1,snr1,'b-d');
grid on;
s = sprintf('Max SQNR = %4.1fdB @ %5.1fdB input\n',maxsnr,inp_maxsnr);
text(-80,maxsnr-10,s);
s = sprintf('Input bin = %3d & BW = %3d\n',f,fB);
text(-80,maxsnr-20,s);
s = sprintf('Max stable input = %4.1fdB',maxstabin);
text(-80,maxsnr-30,s);
xlabel('Input amplitude (dB)');
ylabel('SQNR (dB)');
title('SQNR vs. Input amplitude');


%Discrete to continuous transformation %
ntf1 = filt(b1,a1,Ts);
ntf1 = zpk(ntf1);
% Loop filter computation from NTF %
L1 = filt(1,1,Ts) - inv(ntf1);
[b1,a1]=tfdata(-L1,'v');
%%%%% Compensate for 1 cycle loop delay %%%%%
for i=1:order
b2(i)=b1(i+1);
```

```
end
b2(order+1)=0;
L2 = tf(b2,a1,Ts);
Hs=d2c(L2); % Continuous-time loop filter
```

Function code: ffcoeff

```
function [k w c q] = ffcoeff(Hs)
%Partial fraction expansion of Loop filter with ELD compensation
[nu1,de1]=tfdata(Hs,'v');
cf=nu1(1);
nu2=nu1-nu1(1)*de1;   % Numerators of H'(s)
Hs1=tf(nu2,de1);
[z,p,k]=zpkdata(Hs1,'v');
order=length(nu2)-1;
if order==3
bde1=conv([1 -p(1)],[1 -p(2)]);
de2=[1 -p(3)];
w=[sqrt(bde1(3)) -p(3)];
q=[w(1)/bde1(2)];
[nu1,de1]=tfdata(Hs1,'v');
kc11=nu1(2)*q(1)/w(1);
k1=ceil(kc11);
c1=kc11/k1;
```

```
kc12=(nu1(3)-kc11*w(1)*w(2)/q(1))/w(1)^2;
c2=kc12/k1;
kc23=(nu1(4)-c2*w(1)^2*w(2))/(k1*w(1)^2*w(2));
k2=ceil(kc23);
c3=kc23/k2;
k=[k1 k2];
c=[c1 c2 c3 cf];
end

if order==5
bde1=conv([1 -p(1)],[1 -p(2)]);
bde2=conv([1 -p(3)],[1 -p(4)]);
de3=[1 -p(5)];
w=[sqrt(bde1(3)) sqrt(bde2(3)) -p(5)];
q=[w(1)/bde1(2) w(2)/bde2(2)];
[nu1,de1]=tfdata(Hs1,'v');
kc11=nu1(2)*q(1)/w(1);
kc12=(nu1(3)-kc11*w(1)*w(2)/q(1)/q(2)-kc11*w(1)*w(3)/q(1))/w(1)^2;
k1=ceil(kc11);
c1=kc11/k1;
c2=kc12/k1;
```

```
kc23=(nu1(4)-
(kc11*w(1)*w(2)^2/q(1)+kc11*w(1)*w(2)*w(3)/q(1)/q(2)+kc12*w(1)^2*w(2)/q(2)+kc12*w(1)
^2*w(3)))/k1/w(2)/w(1)^2*q(2);
k2=ceil(kc23);
c3=kc23/k2;
c4=(nu1(5)-
(kc11*w(1)*w(3)/q(1)+kc11*w(1)*w(2)^2/q(1)+kc12*w(1)^2*w(2)^2+kc11*w(1)^2*w(2)*w(3
)/q(2)+k1*kc23*w(2)*w(3)*w(1)^2/q(2)))/k1/k2/w(1)^2/w(2)^2;
kc35=(nu1(6)-
(k1*c2*w(1)^2*w(3)+k1*c2*w(1)^2*w(2)^2+k1*k2*c4*w(2)^2*w(3)*w(1)^2))/k1/k2/w(3)/w(
2)^2/w(1)^2;
k3=ceil(kc35);
c5=kc35/k3;
k=[k1 k2 k3];
c=[c1 c2 c3 c4 c5 cf];
end
```

APPENDIX B

Function Manual: ctloopft

Synopsis: [Hs ntf] = ctloopft(order,OSR,Nlev,Hinf,bw, qf_biquad1,qf_biquad2)

This function first modifies a noise transfer function (NTF) of a discrete-time modulator synthesized by using the Sigma Delta toolbox. It then synthesizes a loop filter for a CT Sigma-Delta modulator from the modified noise transfer function. It automatically produces the bode plot of the modified NTF and SQNR vs. Input amplitude plot of the NTF. This function requires the Sigma-Delta toolbox by Richard Schreier.

Arguments

order	The order of the loop filter
OSR	The oversampling ratio
Nlev	The number of quantization levels
Hinf	The maximum out-of-band gain of the NTF.
bw	The bandwidth frequency of the modulator
qf_biquad1	The finite quality factor of a biquad filter
qf_biquad2	The finite quality factor of a second biquad filter

Output

Hs	The CT loop transfer function.
ntf	The discrete time representation of the loop filter.

Example

A fifth-order low-pass modulator is chosen over 20MHz signal bandwidth with zero optimized for an oversampling ratio of 16, 7 quantization levels, maximum OBG of 3, and Q factors of 4 and 2 respectively.

```
>> [Hs ntf]=ctloopft(5,16,7,3,20e6,4,2)

                    5th-Order Lowpass

Transfer function:

1.944 s^5 + 1.92e009 s^4 + 9.085e017 s^3 + 2.877e026 s^2 + 5.735e034 s + 5.636e042

-----------------------------------------------------------------------------------------------

s^5 + 1.087e008 s^4 + 2.198e016 s^3 + 1.149e024 s^2 + 8.106e031 s + 2.112e039

Zero/pole/gain:

(1-0.95z^-1) (1 - 1.963z^-1 + 0.9737z^-2) (1 - 1.88z^-1 + 0.9122z^-2)

-------------------------------------------------------------------------------

(1-0.5128z^-1) (1 - 1.064z^-1 + 0.3303z^-2) (1 - 1.271z^-1 + 0.642z^-2)

Sampling time (seconds): 1
```

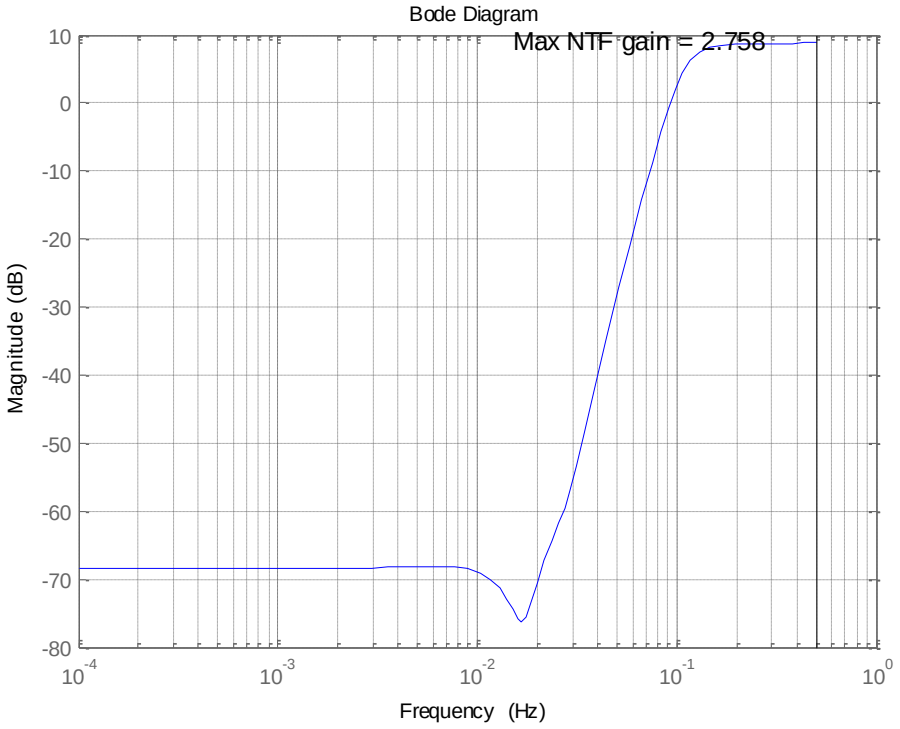

Figure 8. The bode plot of the modified NTF

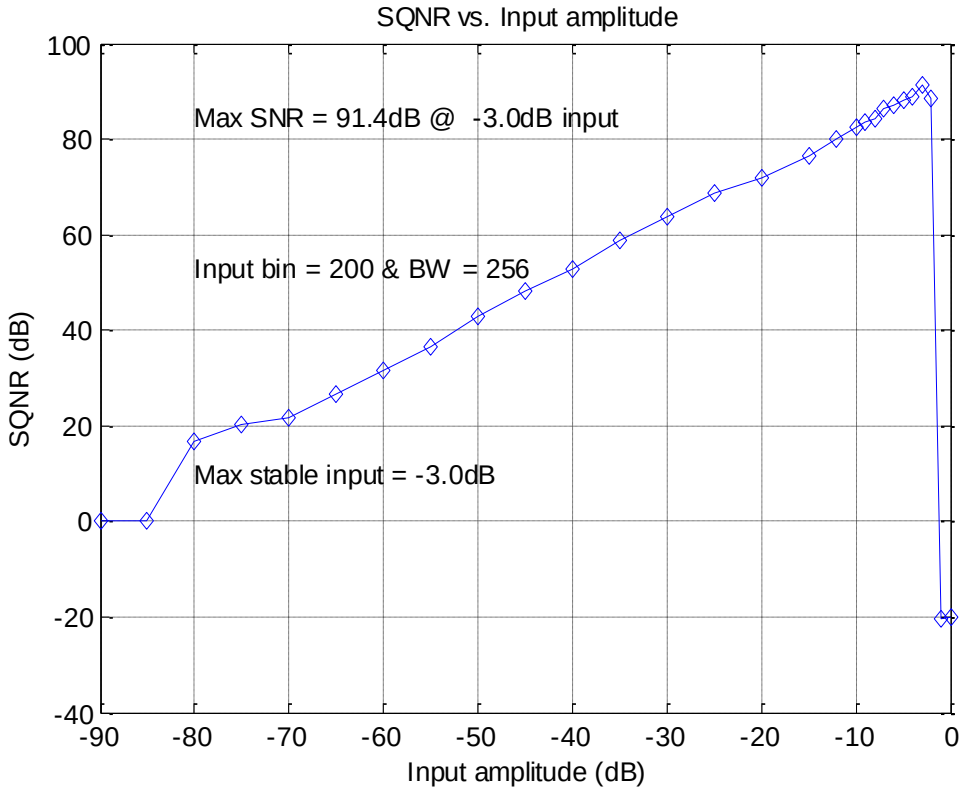

Figure 9. SQNR vs. input amplitude plot of the modified NTF

Function Manual: ffcoeff

Synopsis: [k w c q] = ffcoeff(Hs)

This function calculates the parameter coefficients for feed-forward implementation of CT modulator.

Arguments

Hs	The CT loop transfer function

Output

k	low frequency gains of each stage
w	the cut-off frequencies of each filter stage
c	feed-forward coefficients
q	Q factor(s)

Example

Determine the coefficients for a fifth-order low-pass modulator with feed-forward form;

>> Hs

Transfer function:

1.944 s^5 + 1.92e009 s^4 + 9.085e017 s^3 + 2.877e026 s^2 + 5.735e034 s + 5.636e042

---

s^5 + 1.087e008 s^4 + 2.198e016 s^3 + 1.149e024 s^2 + 8.106e031 s + 2.112e039

>> [k w c] = ffcoeff(Hs)

k =

30 34 2

ω =

1.0e+008 *

1.1761 0.6820 0.3283

c =

0.9684 1.8810 0.9880 0.6939 0.9602 1.9444

q=

4 2